# About Cryptographic Properties of the Qalqan Encryption Algorithm

Nurgul Seilova[1], Almas Kungozhin[1], Renat Ibrayev[1], Lev Gorlov[1], Zhandar Ospanov[1], Ruslan Itemirov[1], and Igor Kiyashko[1]

[1] *Satbayev University, 22 Satbayev str., Almaty,050013, Republic of Kazakhstan*

**Abstract**

Today data confidentiality and privacy are ensured by various encryption algorithms (public key and secret key algorithms). There are no any universal encryption algorithms and different encryption algorithms are implemented to solve different security tasks. Many states have national standards of encryption but Kazakhstan hasn't it today. This paper presents an approach to the development of the Qalqan block symmetric encryption algorithm, taking into account complex cryptographic and operational requirements, as well as the cryptographic properties of the linear and nonlinear blocks of the algorithm that affect its cryptographic strength.

**Keywords**

Cryptography, algorithm, encryption, decryption, linear transformation, nonlinear transformation, key expansion, round transformation.

## 1. Introduction

The Qalqan symmetric block encryption algorithm was developed by the team of the Information Security Research Laboratory of the Institute of Automation and Information Technologies of the KazSRTU named after K.I. Satpayev under the leadership of Ph.D. Nurgul Seilova.

## 2. Algorithm Development Approaches

Claude Shannon defined the basic principles of the reliability of ciphers, the principles of mixing (Confusion) and dispersion (Diffusion), and proposed to build strong cryptographic systems relatively using simple transformations.

Simple transformations make it possible to carry out cryptographic analysis of security with a high level of trust, as well as to guarantee the absence of "backdoors" of various kinds.

The development of the Qalqan block symmetric encryption algorithm was carried out taking into account all the principles of modern cryptography:

1. Using well-studied constructs and transformations.
2. Taking into account trends in the development of cryptography and cryptographic analysis of symmetric block encryption algorithms in order to ensure a "sufficient safety margin."
3. Taking into account the trends in the development of computer technology to avoid the possibility of brute-force attacks and "rainbow tables."
4. Taking into account the peculiarities of the implementation of the Qalqan block symmetric encryption algorithm in the software and hardware-software means of cryptographic information protection.

CPITS-II-2021: Cybersecurity Providing in Information and Telecommunication Systems, October 26, 2021, Kyiv, Ukraine

EMAIL: seilova_na@mail.ru (N. Seilova); kungozhin@gmail.com (A. Kungozhin); ibrayev@gmail.com (R. Ibrayev); gorlov@gmail.com (L. Gorlov); zosp@gmail.com (Z. Ospanov); itemirov@gmail.com (R. Itemirov); lkiyas@gmail.com (I. Kiyashko)
ORCID: 0000-0003-3827-179X (N. Seilova); 0000-0003-6527-1732 (A. Kungozhin); 0000-0002-8882-5677 (R. Ibrayev); 0000-0002-8208-4716 (L. Gorlov); 0000-0003-2685-9174 (Z. Ospanov); 0000-0002-9057-9372 (R. Itemirov); 0000-0002-3250-0996 (I. Kiyashko)

## 3. Qalqan Block Symmetric Encryption Algorithm

### 3.1. Conventions

$V_s$ is the set of all binary strings of length s, where s is a natural number; numbering of substrings and string components is carried out from left to right, starting from zero;

$Blen$ is inlet and outlet block length;

$KLen$ is original encryption key length;

$N$ is number of rounds of encryption algorithm;

$K$ is source encryption key;

$\oplus$ is modulo 2 addition;

$\boxplus$ is modulo $2^{128}$ addition;

$\boxminus$ is modulo $2^{128}$;

$\dotplus$ is modulo 256 addition;

$\dot{-}$ is modulo 256 subtraction.

### 3.2. General Information about the Algorithm

Algorithm architecture: LSX.

Block length: $Blen = 128$ bit.

Key length: $KLen = 256..1024$ bit with 128 bit step.

Number of rounds: $N = 17 + \left\lfloor \frac{KLen-256}{128} \right\rfloor * 2$ (from 17 to 29 rounds).

### 3.3. Parameter Value

### 3.3.1. Nonlinear Bijective Transformation

Nonlinear bijective transformation $S: V_{128} \rightarrow V_{128}$ is specified by substitution $Sbox: V_8 \rightarrow V_8$ (in the following order $S(0), .., S(255)$):

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | eb | 89 | db | cb | f3 | f5 | fb | 90 | e6 | 3d | e5 | 2e | e3 | 0b | 56 | e1 |
| 1 | 6c | 12 | 80 | 28 | ed | 22 | 9 | 4a | ee | 27 | 9b | 58 | 35 | 57 | ef | 94 |
| 2 | 29 | c0 | 16 | 7c | 5e | 87 | 0a | 7e | e8 | 11 | 0e | af | 9a | 84 | 3a | 1a |
| 3 | 69 | 71 | 8c | bc | d2 | 55 | 33 | d1 | 85 | 75 | b5 | 83 | e9 | 50 | 54 | ac |
| 4 | 8a | d6 | 7f | 1f | 14 | 4e | 21 | 82 | 30 | 24 | dd | 9f | 1b | 32 | 20 | a8 |
| 5 | 6a | b0 | 97 | 62 | 19 | d8 | c8 | 0c | 52 | 2 | 5c | 43 | 3 | 95 | 13 | 81 |
| 6 | ab | 77 | a6 | f2 | 59 | 67 | 41 | ec | 76 | 98 | b4 | 73 | 86 | 9c | f7 | cf |
| 7 | dc | ba | a4 | fd | c4 | 99 | df | ce | ea | 1c | 36 | bd | 34 | d7 | 49 | 64 |
| 8 | 5a | 6f | 74 | 1 | a0 | 39 | 91 | 0 | 15 | 3f | 38 | b8 | 8f | 26 | 5f | f8 |
| 9 | 7 | a3 | 0d | da | f0 | e7 | d0 | d9 | 93 | f6 | 6 | 47 | 0f | a1 | 4b | c5 |
| a | 2a | ff | 46 | 60 | d5 | 1d | 2f | a9 | 92 | 17 | 72 | 8e | 7a | aa | 18 | 6e |
| b | 37 | 8 | 1e | 63 | 31 | c2 | bf | c6 | 9e | 65 | d4 | 3b | 96 | 9d | de | 45 |
| c | ca | 2d | a5 | fe | 4d | b9 | 66 | c3 | b3 | cc | ad | 61 | be | 7b | 68 | 88 |
| d | 25 | 2b | 53 | 5b | 44 | 40 | a7 | a2 | 5d | c9 | 51 | ae | e4 | c7 | f9 | 78 |
| e | 70 | cd | 42 | 4f | 4c | 3c | e0 | 3e | 7d | b7 | d3 | b2 | f1 | 8d | 79 | 8b |
| f | 6b | e2 | 10 | 23 | 4 | 6d | c1 | fc | 5 | b6 | f4 | 48 | bb | b1 | 2c | fa |

**Figure 1:** Qalqan S-box

Byte 00 is replaced with eb, 01 with 89, ff with fa (values are in hexadecimal). Substitution is applied to all bytes of the processed block

### 3.3.2. Linear Transformation

Linear function $L: V_{128} \to V_{128}$ converts an 16 byte input block B into an output block R of the same size as follows (bytes are numbered from high to low):

$$\begin{gathered}
R_0 = B_0 \dotplus B_1 \dotplus B_2 \dotplus B_3 \\
R_4 = B_4 \dotplus R_0 \\
R_8 = B_8 \dotplus R_0 \\
R_{12} = B_{12} \dotplus R_0 \\
R_5 = B_4 \dotplus B_5 \dotplus B_6 \dotplus B_7 \\
R_1 = B_1 \dotplus R_5 \\
R_9 = B_9 \dotplus R_5 \\
R_{13} = B_{13} \dotplus R_5 \\
R_{10} = B_8 \dotplus B_9 \dotplus B_{10} \dotplus B_{11} \\
R_2 = B_2 \dotplus R_{10} \\
R_6 = B_6 \dotplus R_{10} \\
R_{14} = B_{14} \dotplus R_{10} \\
R_{15} = B_{12} \dotplus B_{13} \dotplus B_{14} \dotplus B_{15} \\
R_3 = B_3 \dotplus R_{15} \\
R_7 = B_7 \dotplus R_{15} \\
R_{11} = B_{11} \dotplus R_{15}
\end{gathered}$$

### 3.3.3. Key Addition

Round keys are superimposed by modulo $2^{128}$adding with the processed data block (operation $K_{\boxplus}^{i}$, where *i* – round number).

The keys of the first and last rounds are superimposed modulo 2 (operation $K_{\oplus}^{start}$ in the first round and $K_{\oplus}^{fin}$ in the last).

### 3.3.4. Round Transformation

Each round of the $XSL$ algorithm except for the last one, includes sequentially key addition, nonlinear transformation and linear transformation, thus $X_iSL(Text) = L\left(S\left(K_{\boxplus}^{i}(Text)\right)\right)$.

The last round contains only a modulo 2 key overlay operation.

### 3.3.5. Key Expansion

The bytes of the key (starting with the least significant one) are fed in turn to the byte shift registers L0 and L1, starting from L0. Register L0 is 17 bytes long, L1 is 15 bytes long. Thus, a 256-bit key consisting of bytes $\{K_0..K_{31}\}$ fills the registers as follows:

$$L0 = \{K_0, K_2, K_4, K_6, K_8, K_{10}, K_{12}, K_{14}, K_{16}, K_{18}, K_{20}, K_{22}, K_{24}, K_{26}, K_{28}, K_{30}, K_{31}\}$$
$$L1 = \{K_1, K_3, K_5, K_7, K_9, K_{11}, K_{13}, K_{15}, K_{17}, K_{19}, K_{21}, K_{23}, K_{25}, K_{27}, K_{29}\}$$

The feedback of the shift registers is set as follows:

$$L0 = \{L0_1..L0_{16}, S(L0_0) \dotplus L0_1 \dotplus S(L0_3) \dotplus L0_7 \dotplus S(L0_{12}) \dotplus L0_{16}\}$$
$$L1 = \{L1_1..L1_{13}, S(L1_0) \dotplus L1_3 \dotplus S(L1_9) \dotplus L1_{12} \dotplus S(L1_{14})\}$$

The registers move uniformly, after the 17th step each value $L0_{15} \dotplus L1_4$ is filled into the byte of the next round key starting from the least significant.

For keys consisting of more than 256 bits, subsequent bytes in order from least significant to most significant are loaded into shift registers after the 17th step of operation by adding registers modulo 256 to the feedback function starting from register L1. Thus, at each even step of the registers, starting from the 18th, the value of the next byte of the key is superimposed on the feedback value of the L1 register, and at odd steps starting from the 19th - on the feedback value of the L0 register.

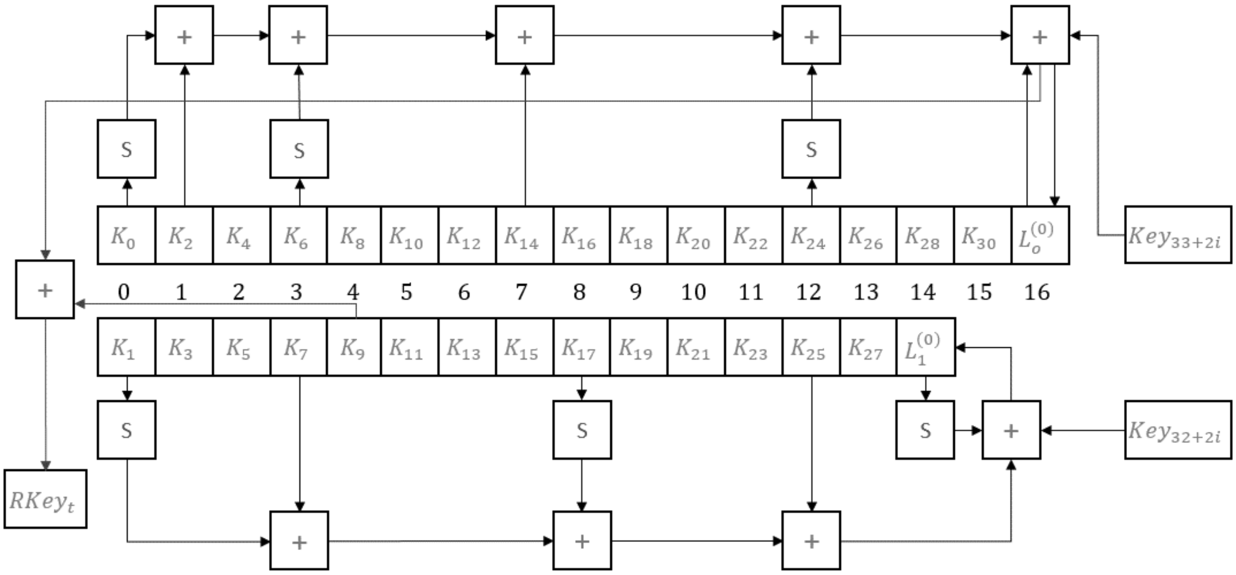
+
S
$K_0$ $K_2$ $K_4$ $K_6$ $K_8$ $K_{10}$ $K_{12}$ $K_{14}$ $K_{16}$ $K_{18}$ $K_{20}$ $K_{22}$ $K_{24}$ $K_{26}$ $K_{28}$ $K_{30}$ $L_o^{(0)}$
$Key_{33+2i}$
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
$K_1$ $K_3$ $K_5$ $K_7$ $K_9$ $K_{11}$ $K_{13}$ $K_{15}$ $K_{17}$ $K_{19}$ $K_{21}$ $K_{23}$ $K_{25}$ $K_{27}$ $L_1^{(0)}$
$Key_{32+2i}$
$RKey_t$

**Figure 2:** Key expansion node.

## 3.4. Encryption Algorithm

### 3.4.1. Encryption

The encryption function consists of three parts and has the form:

$$E_1(Text) = L\left(S\left(K_{\oplus}^{start}(Text)\right)\right)$$

$$E_2(Text) = X_{N-1}SL(\dots X_2SL(X_1SL(E_1(Text))))$$

$$E(Text) = K_{\oplus}^{fin}\left(E_2(Text)\right)$$

The encryption algorithm schematically represented as follows:

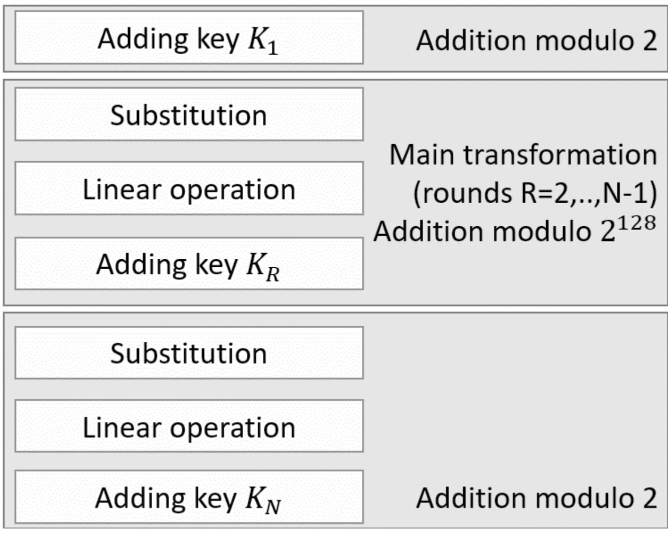

**Figure 3:** General view of the transformation

### 3.4.2. Decryption

For decryption, functions opposite to those described above are used.

Non-linear function InvS is defined by substitution $InvSbox: V_8 \rightarrow V_8$ of the following kind (in the following order $InvS(0), .., InvS(255)$):

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 87 | 83 | 59 | 5c | f4 | f8 | 9a | 90 | b1 | 16 | 26 | 0d | 57 | 92 | 2a | 9c |
| **1** | f2 | 29 | 11 | 5e | 44 | 88 | 22 | a9 | ae | 54 | 2f | 4c | 79 | a5 | b2 | 43 |
| **2** | 4e | 46 | 15 | f3 | 49 | d0 | 8d | 19 | 13 | 20 | a0 | d1 | fe | c1 | 0b | a6 |
| **3** | 48 | b4 | 4d | 36 | 7c | 1c | 7a | b0 | 8a | 85 | 2e | bb | e5 | 9 | e7 | 89 |
| **4** | d5 | 66 | e2 | 5b | d4 | bf | a2 | 9b | fb | 7e | 17 | 9e | e4 | c4 | 45 | e3 |
| **5** | 3d | da | 58 | d2 | 3e | 35 | 0e | 1d | 1b | 64 | 80 | d3 | 5a | d8 | 24 | 8e |
| **6** | a3 | cb | 53 | b3 | 7f | b9 | c6 | 65 | ce | 30 | 50 | f0 | 10 | f5 | af | 81 |
| **7** | e0 | 31 | aa | 6b | 82 | 39 | 68 | 61 | df | ee | ac | cd | 23 | e8 | 27 | 42 |
| **8** | 12 | 5f | 47 | 3b | 2d | 38 | 6c | 25 | cf | 1 | 40 | ef | 32 | ed | ab | 8c |
| **9** | 7 | 86 | a8 | 98 | 1f | 5d | bc | 52 | 69 | 75 | 2c | 1a | 6d | bd | b8 | 4b |
| **a** | 84 | 9d | d7 | 91 | 72 | c2 | 62 | d6 | 4f | a7 | ad | 60 | 3f | ca | db | 2b |
| **b** | 51 | fd | eb | c8 | 6a | 3a | f9 | e9 | 8b | c5 | 71 | fc | 33 | 7b | cc | b6 |
| **c** | 21 | f6 | b5 | c7 | 74 | 9f | b7 | dd | 56 | d9 | c0 | 3 | c9 | e1 | 77 | 6f |
| **d** | 96 | 37 | 34 | ea | ba | a4 | 41 | 7d | 55 | 97 | 93 | 2 | 70 | 4a | be | 76 |
| **e** | e6 | 0f | f1 | 0c | dc | 0a | 8 | 95 | 28 | 3c | 78 | 0 | 67 | 14 | 18 | 1e |
| **f** | 94 | ec | 63 | 4 | fa | 5 | 99 | 6e | 8f | de | ff | 6 | f7 | 73 | c3 | a1 |

**Figure 4:** Qalqan inverted S-box

Inverse linear operation $InvL: V_{128} \rightarrow V_{128}$ converts 16 byte input block B to output block R of the same size as follows (bytes are numbered in ascending order):

$$R_1 = B_1 \dot{-} B_5$$
$$R_2 = B_2 \dot{-} B_{10}$$
$$R_3 = B_3 \dot{-} B_{15}$$
$$R_4 = B_4 \dot{-} B_0$$
$$R_6 = B_6 \dot{-} B_{10}$$
$$R_7 = B_7 \dot{-} B_{15}$$
$$R_8 = B_8 \dot{-} B_0$$
$$R_9 = B_9 \dot{-} B_5$$
$$R_{11} = B_{11} \dot{-} B_{15}$$
$$R_{12} = B_{12} \dot{-} B_0$$
$$R_{14} = B_{14} \dot{-} B_0$$
$$R_0 = B_0 \dot{-} R_1 \dot{-} R_2 \dot{-} R_3$$
$$R_5 = B_5 \dot{-} R_4 \dot{-} R_6 \dot{-} R_7$$
$$R_{10} = B_{10} \dot{-} R_8 \dot{-} R_9 \dot{-} R_{11}$$
$$R_{15} = B_{15} \dot{-} R_{12} \dot{-} R_{13} \dot{-} R_{14}$$

The round keys $K_{\boxminus}$ are superimposed by modulo $2^{128}$subtraction. The keys of the first and last rounds are superimposed by modulo 2 operation.

One round of the $SLX$ decryption algorithm includes sequentially linear, non-linear transformations and key addition, thus $SLX_i(Cipher) = K_{\boxminus}^{i}(InvL(InvS(Cipher)))$.

The decrypting operation of the *Cipher* ciphertext on round keys *Key* is represented as follows:

$$D_1(Cipher) = K_{\oplus}^{fin}(Cipher)$$

$$D_2(Cipher) = SLX_1(\dots SLX_{N-2}(SLX_{N-1}(D_1(Cipher))))$$
$$D(Cipher) = K_{\oplus}^{start}\left(InvL\left(InvS\big(D_2(Cipher)\big)\right)\right)$$

## 4. Properties of the Nonlinear Node of Algorithm

The correct choice of the characteristics of vector Boolean functions is the main factor in ensuring security, since they are the only non-linear node of the symmetric block encryption algorithm.

In this paper, the main cryptographic properties of the nonlinear node of the Qalqan symmetric block encryption algorithm, implemented as a vector Boolean function of 8 variables are investigated.

Due to the existence of serious claims to the secret internal structure of the nonlinear node of the block symmetric encryption algorithm Kuznechik and the hash function Stribog [15], the nonlinear node of the block symmetric encryption algorithm Qalqan was designed according to the principle of maximum transparency.

For this, a similar to that used in the AES block symmetric encryption algorithm, the generation of a replacement table according to proposed by K. Nyberg in 1991 [16] method was chosen. The constants used in the generation have been changed in order to achieve closer to the optimal values [17] of such cryptographically important parameters like lowering the maximum of the differential profile in addition and XOR (counteraction to differential analysis), lowering the maximum of the linear approximations table (counteraction to linear analysis), high algebraic degree, maximum to the class of affine transformations distance (high nonlinearity), absence of linear structures.

As part of the research of the main cryptographic properties, the following properties shown in Table 1 were established.

**Table 1**

Research results of the main cryptographic properties of substitution nodes affecting cryptographic strength

| № | Vector Boolean function property | Qalqan symmetric block encryption algorithm |
|---|---|---|
| 1 | Algebraic degree of coordinate functions | 7 |
| 2 | Balance of coordinate functions | $wt(T_{f_i}) = 2^{8-1} = 128$, $f_i$-coordinate boolean function, $i = 1..8$ |
| 3 | Perfect poise | $wt(T_{f_i}) = 2^{8-1} = 128$, $f_i$-coordinate boolean function, $i = 1..8$ |
| 4 | Avalanche criterion | $2^{8-1}$±16 |
| 5 | Correlation immunity | 0 |
| 6 | Nonlinearity, distance to a class of affine functions | 112 |
| 7 | Algebraic immunity | $4 \geq AI(f) > 1$ |
| 8 | Linear structures | Absent |
| 9 | Differential XOR characteristic | 4 |
| 10 | Differential addition characteristic | 8 |
| 11 | Absolute maximum of the table of linear approximations | 32 |

## 5. Properties of a Linear Node of the Algorithm

The linear transformation of the Qalqan symmetric block cipher algorithm is built on a SQUARE-like architecture and is a series of modulo 256 addition operations that propagate the influence of the bytes on each line of the intermediate state to the corresponding column. If we represent the state of the processed block in the form of a 4x4 byte matrix, then during the conversion to a byte located on the main diagonal (let's call it a diagonal byte), the remaining three bytes of this row are modulo 256summed, then the resulting sum is superimposed by adding modulo 256 to the rest three bytes located on the same column as the diagonal byte. Row and column additions occur independently of each other's results, i.e. changes in the state of the processing block are made immediately after all operations (figure 5).

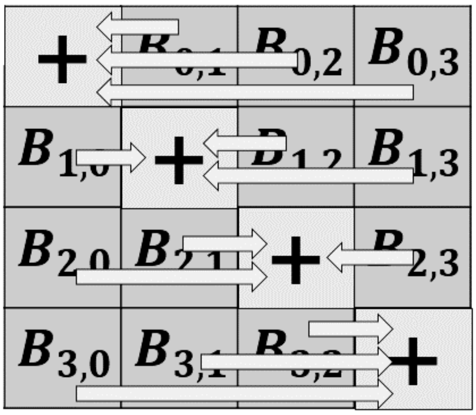

**Figure 5:** Linear operation of the Qalqan algorithm, stage 1 (absorption)

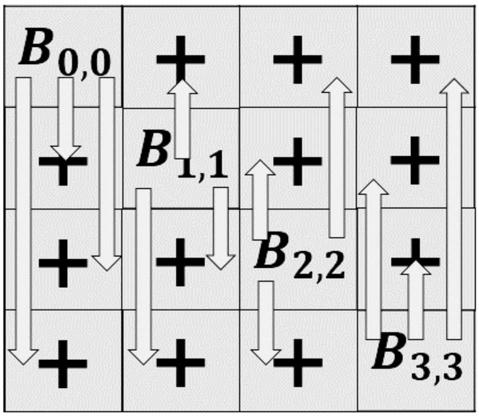

**Figure 6:** Linear operation of the Qalqan algorithm, stage 2 (spreading).

To conclude about the effectiveness of the transformation, a comparison of the main characteristics with similar transformations of the symmetric block encryption algorithms AES [22] and Kuznechik [23] was made. Both algorithms are widely used and well enough studied to be considered as cryptographically strong, and are similar in main parameters to the Qalqan algorithm.

**Table 2**
Comparison of the main parameters of the Qalqan, AES and Kuznechik algorithms

| Parameter | Qalqan | AES | Kuznechik |
|---|---|---|---|
| Key length | 256..1024 | 128/192/256 | 256 |
| Block length | 128 | 128 | 128 |
| Architecture | SP-network, SQUARE | LSX, SQUARE | SP-network, LSX |
| Number of rounds | 10/12/14 | 17..29 | 10 |
| Substitution node input | 8 bits | 8 bits | 8 bits |

The difference between the Qalqan block symmetric encryption algorithm from most of the known algorithms is key overlay with bit transfer, i.e. instead of the XOR operation, modulo $2^{128}$ addition is used in the middle rounds of the algorithm. This operation has a positive effect on the complication of the relationship between the bits of encrypted data and increases the resistance to a number of modern attacks.

In this work, for the sake of purity of comparison, this feature of the algorithm will not be taken into account, since in the compared algorithms the key is overlaid by means of the XOR operation.

According to the principles of K. Shannon [14], the main purpose of the linear node is to mix the bits of the processed data block, which entails the spread of the entropy of the key bits over the entire data block, as well as the mutual influence of the bits of the processed block itself. The efficiency of this influence is traditionally estimated by calculating the "avalanche effect" [13].

Ideally, when one bit of the key or data changes, half (on average) of the remaining bits of the block being processed after the linear operation should change. It is possible to construct such a block in practice (for example, using a linear shift register), but this is rarely used due to the high computational complexity of such transformation. In general, bit operations in modern processors and microcontrollers are relatively difficult to implement [20] in comparison with byte operations; therefore, it is advisable to consider byte-oriented transformations, which, nevertheless, are capable of ensuring the mutual influence of bits.

To provide the required avalanche effect, each byte must have an effect on the remaining bytes, while it is desirable to be different for bytes in different positions. The operations used in this case should also be significantly efficient in terms of memory and computational complexity, as well as being possible to implement on the maximum number of modern hardware platforms.

Measurement of the avalanche effect of linear transformations of the Qalqan, AES and Kuznechik algorithms for one, two and three rounds of the algorithm simulating the operation of an SP-network showed the following results:

**Table 3**

Avalanche effect of linear transformation of algorithms Qalqan, AES and Kuznechik

| Number of rounds | Qalqan | AES | Kuznechik |
|---|---|---|---|
| 1 | 8.609375 | 5.750000 | 62.554688 |
| 2 | 65.765625 | 66.500000 | 63.742188 |
| 3 | 64.156250 | 63.953125 | 63.882812 |

To simulate the operation of several rounds of the symmetric block encryption algorithm an overlay of fixed bytes obtained using the standard pseudo-random number generator of the stdlib.h library between linear operations and byte substitution used in the Qalqan algorithm were added.

The measurement of the execution time of linear operations for the algorithms Qalqan, AES and Kuznechik has been carried out. For this, linear transformations of these algorithms are implemented and their speed is compared. Since each individual transformation is performed quickly, the measurement was made for large (on the order of hundreds of thousands) series of transformations separately with enabled and disabled optimization. Results are presented in the following tables:

**Table 4**

Execution speed of linear transformation of algorithms Qalqan, AES and Kuznechik (time for 20 million operations, μs)

| Optimization | Qalqan | AES | Kuznechik |
|---|---|---|---|
| None | 44750.1 | 100933.7 | 622452.7 |
| Full | 11863.5 | 34765.4 | 163702.3 |

The linear transformation of the Kuznechik algorithm for time optimization requires preliminary initialization of 9 tables of 256 bytes each.

## 6. Conclusions

Thus, it has been proven that the speed of linear transformation is significantly higher than world analogues. Moreover, as in the AES algorithm, there is a possibility of parallel implementation of the conversion into four threads, as well as implementation in single (one) clock cycle on FPGA [21].

It should be noted the simplicity of the implementation of the linear transformation of the Qalqan algorithm, since it contains only modulo 256 addition operations. Thus, when operating with 8-bit variables, the programmer uses only the addition operation.

## 7. References

[1] Fomichev V.M. Methods of discrete mathematics in cryptology. M.: Dialogue - MIFI, 2010. 424p.

[2] Romanko D.A., Fomichev V.M. Methods of constructing cryptographic generators with a given non-repetition index of the output sequences, Applied discrete mathematics. Application. 2016. #9. pp. 65-67.

[3] B. Schneier. M.: Triumph, Applied cryptography: Protocols, algorithms and source texts in C, 2002. 610 p.

[4] J.M. Alfred, P.C. van Oorschot, S.A. Vanstone, Handbook of Applied Cryptography (Discrete Mathematics and Its Applications), CRC Press, 1996. P. 796.

[5] Pudovkina M. A. Related-Key Attack on Block Ciphers with Weak Recurrent Key Schedules, Foundations and Practice of Security. Berlin: Springer-Verlag, 2011, pp. 90–101.

[6] Isobe T.A Single-Key Attack on the Full GOST Block Cipher, Journal of Cryptology, 2013, no. 26, pp. 172–189.

[7] Astashkina E.N., Lysenko I.V. An Approach to Formation of the Key Schedule for a Block Symmetric Cryptoalgorithm GOST 28147–89. Information processing systems. 2010. #6. pp. 30–34.

[8] May L. Strengthening the Key Schedule of the AES. / L. May, M. Henricksen, W. Millan, G. Carter, E. Dawson // Information Security and Privacy. Berlin: Springer-Verlag, 2002. P. 226–240.

[9] Fomichev V.M. On the key schedule of block ciphers without weak keys, Tomsk State University. Applied discrete mathematics. Application #9. pp. 70–73.

[10] E. Biham, New types of cryptanalytic attacks using related keys, Springer-Verlag, 1994. DOI:10.1007/BF00203965

[11] A. Biryukov, D. Khovratovich, Related-Key Cryptanalysis of the Full AES-192 and AES-256, Springer Berlin Heidelberg, 2009. DOI:10.1007/978-3-642-10366-7_1

[12] Gorlov L., Ibraev R. Candidate for national standards of the Republic of Kazakhstan - Qalqan encryption algorithm https://www.ruscrypto.ru/resource/archive/rc2021/files/02_gorlov_ibrayev.pdf

[13] H. Feistel, «Cryptography and Computer Privacy / Scientific American, Vol. 228, No. 5, 1973.

[14] Shannon K. Communication Theory of Secrecy Systems / Bell System Technical Journal, 1949.

[15] L. Perrin, A. Udovenko, Exponential S-Boxes: a Link Between the S-Boxes of BelT and Kuznyechik/Streebog, IACR Transactions on Symmetric Cryptology, 2016, P. 99–124, ISSN 2519-173X, doi:10.13154/tosc.v2016.i2.99-124.

[16] Kaisa Nyberg, Perfect non-linear S-boxes // EUROCRYPT 1991: Advances in Cryptology — EUROCRYPT ’91, P. 378-386.

[17] Jie Cui, Liusheng Huang, Hong Zhong, Chinchen Chang, Wei Yang, An improved AES S- box and its performance analysis // International Journal of Innovative Computing, Information and Control, 2011, vol. 7, #5(A), P. 2291-2302.

[18] E. Biham and A. Shamir, Differential cryptoanalysis of DES with a reduced number of rounds // Advanced Crypto, 1990.

[19] Matsui M. Linear Cryptanalysis Method for DES Cipher // Advances in Cryptology — EUROCRYPT ’93: Workshop on the Theory and Application of Cryptographic Techniques Lofthus, Norway, May 23–27, 1993 Proceedings / T. Helleseth, Berlin: Springer Berlin Heidelberg pp, 1993, P. 386-397, 465 p., ISBN 978-3-540-57600-6, doi:10.1007/3-540-48285-7_33

[20] Knut D. The art of programming. Vol. 1. Basic algorithms / Moscow: Williams, 2002, T. 1, p. 720.

[21] Harshali Zodpe, Ashok Sapkal, «An efficient AES implementation using FPGA with enhanced security features», Journal of King Saud University - Engineering Sciences, Volume 32, Issue 2, 2020, P. 115-122.

[22] Federal Information Processing Standards Publication 197 November 26, 2001 Specification for the Advanced Encryption Standard (AES) / http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf (15.08.2021).

[23] GOST R 34.12-2015. Block Cipher ”Magma”

[24] Iavich M., Gagnidze A., Iashvili G., Gnatyuk S., Vialkova V. Lattice based Merkle, CEUR Workshop Proceedings, Vol. 2470, pp. 13-16, 2019.

[25] B. Akhmetov, S. Gnatyuk, V. Kinzeryavyy, Kh.Yubuzova, Studies on practical cryptographic security analysis for block ciphers with random substitutions, International Journal of Computing, Vol. 19, Issue 2, pp. 298-308, 2020.